IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Appellants:	Gerard Chauvel, et al.	§	Confirmation No.:	9644
		§		
Serial No.:	10/631,939	§		
		§		
Filed:	July 31, 2003	§		
		§	Group Art Unit:	2183
		§		
For:	A Multi-Processor Computing System Having A JAVA Stack Machine and a RISC-Based Processor	§		
		§	Examiner:	A. J. Petranek
		§		

APPEAL BRIEF

Mail Stop Appeal Brief – Patents
Commissioner for Patents
PO Box 1450
Alexandria, VA 22313-1450

Date: November 16, 2007
Atty. Docket No.: TI-35710 (1962-05423)

Sir:

Appellants hereby submit this Appeal Brief in connection with the above-identified application. A Notice of Appeal is filed concurrently herewith.

TABLE OF CONTENTS

I. REAL PARTY IN INTEREST

The real party in interest is Texas Instruments Inc., a Delaware corporation, having its principal place of business in Dallas, Texas. The Assignment from the inventors to Texas Instruments-France was recorded on December 8, 2003 at Reel/Frame 014189/0772 and from Texas Instruments-France to Texas Instruments Inc. was recorded on March 12, 2004 at Reel/Frame 014421/0954.

II. RELATED APPEALS AND INTERFERENCES

Appellants would like to make the Board aware of a related appeal on a co-pending U.S. Patent Application No. 10/632,024.

III. STATUS OF THE CLAIMS

Originally filed claims:	1-19.
Claim cancellations:	3-4, 8-9, 12-13, 15-16, 18-19.
Added claims:	20-24.
Presently pending claims:	1-2, 5-7, 10-11, 14, 17, 20-24.
Allowed claims:	None.
Presently appealed claims:	1-2, 5-7, 10-11, 14, 17, 20-24.

IV. STATUS OF THE AMENDMENTS

No claims were amended after the final Office action dated August 17, 2007.

V. SUMMARY OF THE CLAIMED SUBJECT MATTER

The specification is directed to a multi-processor computing system having a Java stack machine and a RISC-based processor.[1] At least some of the various embodiments are systems as in claim 1

A system, comprising:
a first processor[2];
a second processor coupled to the first processor, the second processor having a core and comprising stack storage residing in the core[3];
memory coupled to, and shared by, the first and second processors[4]; and
a synchronization unit coupled to the first and second processors, said synchronization unit synchronizes the execution of the first and second processors[5];
wherein the second processor executes stack-based instructions while the first processor executes one or more tasks wherein the first processor manages the memory via an operating system that executes only on the first processor and the first processor executes a virtual machine that controls the execution of a program on the second processor[6];
wherein the first processor executes a transaction targeting a pre-determined address and the synchronization unit detects said pre-determined address and asserts a wait signal to cause said first processor to enter a reduced power or reduced performance mode[7]; and
wherein said second processor asserts a wait release signal that is received by said synchronization unit and that causes said synchronization unit to deassert said wait signal to the first processor.[8]

---

[1] Due to typographical error in the specification the paragraph numbers are restarted in the middle of specification. Thus, to avoid confusion the published version of the application as filed, US Patent Application Publication No. 2004/0078850, is used to provide support for the claims. US Patent Application Publication No. 2004/0078850 Title.

[2] US Patent Application Publication No. 2004/0078850 page 2, paragraph [0024], lines 6-8 within the paragraph. The balance of this Appeal Brief uses a shorthand notation for citations to the US Patent Application Publication No. 2004/0078850 in the form: ([page],[paragraph number], [lines]). Thus, this illustrative citation in the shorthand form reads (2, [0024], lines 6-8). *See also*, Figures 1, element 104.

[3] (2, [0024], lines 1-6); Figure 2, elements 120, 146.

[4] (2, [0024], lines 10-13); Figure 1, element 106.

[5] (5, [0046], lines 1-8); Figure 6, element 206.

[6] (2, [0025], lines 1-17); Figure 1, element 104.

[7] (5, [0045], lines 1-19); Figure 6, element 206.

[8] (5, [0047], lines 1-10); Figure 6, element 206.

Other illustrative embodiments are methods as in claim 10

A method, comprising:
synchronizing the execution of first and second processors[9], the second processor having a core and comprising stack storage residing in the core[10], wherein synchronizing comprises detecting that the first processor is executing a transaction targeting a pre-determined address and asserting a wait signal to cause said first processor to enter a reduced power or reduced performance mode[11] and synchronizing further comprises the second processor causing the wait signal to be de-asserted to terminate the first processor's reduced power or reduced performance mode[12];
executing stack-based instructions in the second processor while the first processor executes one or more tasks; executing an operating system on the first processor and not on the second processor; executing a virtual machine on the first processor that controls the execution of a program on the second processor[13]; and
the first processor managing memory accessible to both the first and second processors via the operating system.[14]

Yet still other illustrative embodiments are systems as in claim 17:

A system, comprising:
a first processor[15];
a second processor coupled to the first processor, the second processor having a core and comprising stack storage residing in the core and having an internal data memory that holds a contiguous block of memory defined by an address stored in a register, and wherein local variables are stored in said data memory[16];
memory coupled to, and shared by, the first and second processors[17]; and
a synchronization unit coupled to the first and second processors[18], said synchronization unit asserts a first signal to the first processor to

[9] (5, [0046], lines 1-8); Figure 6, element 206.

[10] (2, [0024], lines 1-6); Figure 1, element 102.

[11] (5, [0045], lines 1-19); Figure 6, element 206.

[12] (5, [0047], lines 1-10); Figure 6, element 206.

[13] (2, [0025], lines 1-17); Figure 1, element 104.

[14] (2, [0024], lines 10-13); Figure 1, element 106.

[15] (2, [0024], lines 6-8); Figure 1, element 104.

[16] (3, [0029], lines 1-16); Figure 3, 120.

[17] (2, [0024], lines 10-13); Figure 1, element 106.

[18] (5, [0046], lines 1-8); Figure 6, element 206.

cause the first processor to cease executing instructions[19] and said synchronization unit receives a second signal from the second processor which thereby causes the synchronization unit to de-assert the first signal[20];

wherein the second processor executes stack-based instructions while the first processor executes one or more tasks wherein the first processor manages the memory via an operating system that executes only on the first processor and the first processor executes a virtual machine that controls the execution of a program on the second processor.[21]

[19] (5, [0045], lines 1-19); Figure 6, element 206.

[20] (5, [0047], lines 1-10); Figure 6, element 206.

[21] (2, [0025], lines 1-17); Figure 1, element 104.

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Whether claims 1-2, 5-7, 10-11, 14, 17, 20-21 and 23 are obvious under 35 USC § 103 over Feierbach et al. (U.S. Pat. No. 6,088,786, hereinafter Feierbach) in view of Seal et al. (U.S. Pat. No. 6,965,984, hereinafter Seal) further in view of Kloth (U.S. Pat. No. 6,549,961).

VII. ARGUMENT

A. Section §103 Rejections over Feierbach, Seal and Kloth

1. Claims 1-2, 5-7, 10-11, 14, 17, 20-21 and 23

Claims 1-2, 5-7, 10-11, 14, 17, 20-21 and 23 stand rejected as allegedly obvious over Feierbach, Seal and Kloth. Claim 1 is representative of this grouping of claims. The grouping should not be construed to mean the patentability of any of the claims may be determined in later actions (*e.g.*, actions before a court) based on the groupings. Rather, the presumption of 35 U.S.C. § 282 shall apply to each of these claims individually.

Feierbach is directed to coupling a stack based processor to a register based functional unit.[22] Feierbach teaches a processor that comprises a stack processor, a register processor and a copy unit.[23] The stack processor is configured to receive stack instructions from the instruction decoder and execute the received instructions.[24] The "register processor" executes "extended stack instructions" using the operands copied to the register file from the stack by the copy-unit.[25] In fact, "the register processor" executes the "extended stack instructions" as provided by the stack processor.[26] Thus, Feierbach teaches a stack processor and a "register processor" that each execute stack instructions, but Feierbach is silent as to a processor that executes a virtual machine in control of a separate stack processor. Moreover, Feierbach does not expressly discuss the specifics of the "register processor", but instead merely incorporates by reference a "register processor" as disclosed by Yung (U.S. Pat. No. 5,996,066).[27]

Representative claim 1, by contrast, specifically recites, "a first processor; … the first processor executes a virtual machine that controls the execution of a program on the second processor." Appellants respectfully submit that Feierbach, Seal and Kloth fail to teach or fairly suggest such a system. The Office Action takes the position that the

---

[22] Feierbach Title.

[23] Feierbach Col. 6, lines 53-59.

[24] Feierbach Col. 7, lines 27-37.

[25] Feierbach Col. 11, lines 45-53.

[26] Feierbach Col. 8, lines 21-32.

[27] *See* Office Action of August 17, 2007, page 3, last paragraph.

claimed "first processor" corresponds to the register processor 204 in Feierbach.[28] However, in Feierbach the register processor 204 only executes extended stack instructions as provided by Feierbach's stack processor. Feierbach fails to teach or suggest that the register processor in any way controls the Feierbach stack processor. In fact, it is quite the opposite. Thus, even if hypothetically the teachings of Seal and Kloth are precisely as the Office action suggests (which Appellants do not admit), the references still fail to teach or fairly suggest "a first processor; … the first processor executes a virtual machine that controls the execution of a program on the second processor." For this reason alone the rejections of this grouping should be reversed and the claims set for issue.

Moreover, representative claim 1 recites, "a second processor coupled to the first processor…wherein the second processor executes stack-based instructions while … the first processor executes a virtual machine that controls the execution of a program on the second processor." The Office action relies on Feierbach's stack processor for the claimed second processor that "executes stack-based instructions"; however, in paraphrasing the claim the Office action misquotes the claim to be "the second processor executes a virtual machine that controls the execution of a program on the second processor",[29] and then the Office action relies on Feierbach's stack processor as the processor that executes the virtual machine.[30] Stated otherwise, the Office action improperly uses Feierbach's stack processor as the claimed second processor that "executes stack-based instructions" and the processor that controls the second processor. Thus, even if hypothetically the teachings of Seal and Kloth are precisely as the Office action suggests (which Appellants do not admit), the references still fail to teach or fairly suggest "a second processor coupled to the first processor…wherein **the second processor executes stack-based instructions** while … **the first processor** executes a virtual machine that controls the execution of a program on **the second processor**." For

[28] Office Action of August 17, 2007, page 2, seventh paragraph.

[29] Office Action of August 17, 2007, page 3, last paragraph.

[30] Office Action of August 17, 2007, page 3, last paragraph.

these additional reasons the rejections of this grouping should be reversed and the claims set for issue.

Kloth is directed to semaphore access in a multiprocessor system.[31] In Kloth, a processor requests to access a resource; this request, in turn, is forwarded to a bridge which determines whether to grant the access to the resource.[32] The data of the request is sent to the bridge within a data access request over a processor bus.[33] The bridge examines a semaphore for the requested resource, and determines if the requested resource is available, and if the requested resource is not available then a halt signal is asserted to the requesting processor.[34]

Representative claim 1, by contrast, recites "wherein the first processor executes a transaction targeting a pre-determined address and the synchronization unit detects said pre-determined address and asserts a wait signal to cause said first processor to enter a reduced power or reduced performance mode." Appellants respectfully submit that Feierbach, Seal and Kloth fail to teach or fairly suggest such a system. Kloth teaches a system where the processor wanting to access a resource sends a request to a bridge to determine if the resource is available. Stated otherwise, Kloth teaches a system where the requesting processor is halted by the bridge; but, the trigger for halting the requesting processor is the data within the data access request sent over the processor bus and unavailability of the requested resource. Thus, even if hypothetically the teachings of Feierbach and Seal are precisely as the Office action suggests (which Appellants do not admit), the references still fail to teach or fairly suggest "wherein the **first processor executes a transaction targeting a pre-determined address** and **the synchronization unit detects said pre-determined address** and asserts a wait signal to cause said first processor to enter a reduced power or reduced performance mode."

Based on the foregoing, Appellants respectfully submit that the rejections of the claims of this grouping should be reversed, and the claims set for issue.

---

[31] Kloth Title.

[32] Kloth Col. 3, line 51 to col. 4, line 2.

[33] Kloth Col. 3, lines 56-57.

[34] Kloth Col. 3, line 57 to Col. 4, line 2.

2. Claim 22 and 24

Claim 22 and 24 stands rejected as allegedly obvious over Feierbach, Seal, Kloth and Evoy. Claims 22 and 24 are allowable for at least the same reasons as delineated in Section VII(A)(1).

B. Conclusion

For the reasons stated above, Appellants respectfully submit that the Examiner erred in rejecting all pending claims. It is believed that no extensions of time or fees are required, beyond those that may otherwise be provided for in documents accompanying this paper. However, in the event that additional extensions of time are necessary to allow consideration of this paper, such extensions are hereby petitioned under 37 C.F.R. § 1.136(a), and any fees required (including fees for net addition of claims) are hereby authorized to be charged to the Texas Instruments, Inc. Deposit Account No. 20-0668.

Respectfully submitted,

/Utpal D. Shah/

Utpal D. Shah
PTO Reg. No. 60,047
CONLEY ROSE, P.C.
(512) 610-3440 (Phone)
(512) 610-3456 (Fax)
AGENT FOR APPELLANTS

VIII. CLAIMS APPENDIX

1. (Previously Presented) A system, comprising:

a first processor;

a second processor coupled to the first processor, the second processor having a core and comprising stack storage residing in the core;

memory coupled to, and shared by, the first and second processors; and

a synchronization unit coupled to the first and second processors, said synchronization unit synchronizes the execution of the first and second processors;

wherein the second processor executes stack-based instructions while the first processor executes one or more tasks wherein the first processor manages the memory via an operating system that executes only on the first processor and the first processor executes a virtual machine that controls the execution of a program on the second processor;

wherein the first processor executes a transaction targeting a pre-determined address and the synchronization unit detects said pre-determined address and asserts a wait signal to cause said first processor to enter a reduced power or reduced performance mode; and

wherein said second processor asserts a wait release signal that is received by said synchronization unit and that causes said synchronization unit to deassert said wait signal to the first processor.

2. (Original) The system of claim 1 wherein the second processor comprises an internal data memory that holds a contiguous block of memory defined by an address stored in a register, and wherein local variables are stored in said data memory.

3. – 4. (Cancelled).

5. (Original) The system of claim 1 wherein the stack-based instructions comprise Java bytecodes and the first processor comprises a RISC processor so that the RISC processor executes one or more tasks while the second processor executes Java code.

6. (Original) The system of claim 1 further including a main stack residing outside the second processor's core and coupled to the stack storage in the second processor's core.

7. (Original) The system of claim 6 wherein the stack storage in the second processor's core provides an operand to execute a stack-based instruction in the second processor.

8.-9. (Canceled)

10. (Previously Presented) A method, comprising:

synchronizing the execution of first and second processors, the second processor having a core and comprising stack storage residing in the core, wherein synchronizing comprises detecting that the first processor is executing a transaction targeting a pre-determined address and asserting a wait signal to cause said first processor to enter a reduced power or reduced performance mode and synchronizing further comprises the second processor causing the wait signal to be de-asserted to terminate the first processor's reduced power or reduced performance mode;

executing stack-based instructions in the second processor while the first processor executes one or more tasks;

executing an operating system on the first processor and not on the second processor;

executing a virtual machine on the first processor that controls the execution of a program on the second processor; and

the first processor managing memory accessible to both the first and second processors via the operating system.

11. (Original) The method of claim 10 further including storing local variables in an internal data memory in the second processor, the internal data memory configured to store a contiguous block of memory defined by an address stored in a register.

12. – 13. (Cancelled).

14. (Original) The method of claim 11 further comprising providing a main stack residing outside the second processor's core and providing an operand from the stack storage in the second processor's core and executing a stack-based instruction in the second processor using the operand.

15.-16. (Canceled)

17. (Previously Presented) A system, comprising:

a first processor;

a second processor coupled to the first processor, the second processor having a core and comprising stack storage residing in the core and having an internal data memory that holds a contiguous block of memory defined by an address stored in a register, and wherein local variables are stored in said data memory;

memory coupled to, and shared by, the first and second processors; and

a synchronization unit coupled to the first and second processors, said synchronization unit asserts a first signal to the first processor to cause the first processor to cease executing instructions and said synchronization unit receives a second signal from the second processor which thereby causes the synchronization unit to de-assert the first signal;

wherein the second processor executes stack-based instructions while the first processor executes one or more tasks wherein the first processor manages the memory via an operating system that executes only on the first processor and the first processor executes a virtual machine that controls the execution of a program on the second processor.

18. – 19. (Cancelled).

20. (Previously presented) The system of claim 1 wherein a clock internal to the first processor is disabled thereby effectuating the reduced power or reduced performance mode.

21. (Previously Presented) The system of claim 1 wherein said wait signal remains asserted until said synchronization unit deasserts said wait signal.

22. (Previously Presented) The system of claim 1 wherein said second processor asserts said wait release signal when said second processor requires support from said first processor.

23. (Previously Presented) The system of claim 17 wherein said synchronization unit continues to assert said first signal until the synchronization unit either the synchronization unit receives said second signal from the second processor or the synchronization unit receives an interrupt signal.

24. (Previously Presented) The system of claim 17 wherein said second processor asserts said second signal when said second processor requires support from said first processor.

IX. EVIDENCE APPENDIX

None.

X. RELATED PROCEEDINGS APPENDIX

None.